Interrupts, Exceptions, and System Calls

Andre M. Maier, DHBW Ravensburg
dhbw@andre-maier.com

Why do we need this?

- A CPU needs mechanisms to react on internal or external events that may need immediate attention.
- Some examples
  - The user moves the mouse
  - The user presses a key
  - A network packet arrives
- Problem: The CPU may be busy running a program.

Andre M. Maier
Dept. of Computer Science

Polling

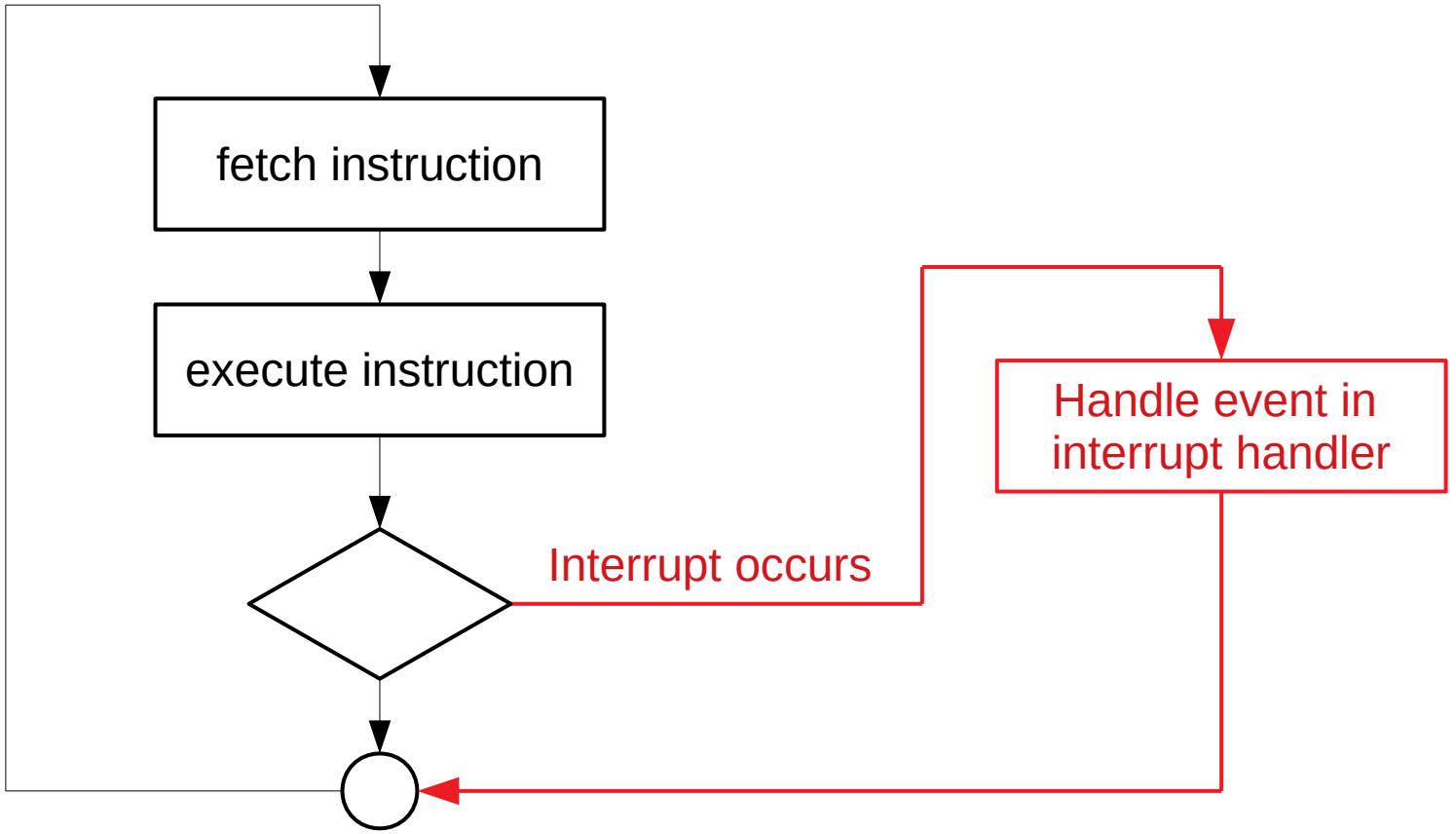
- Operating system periodically checks each potential source (keyboard, mouse, ...) if an event has occurred
- Analogy: Assume your doorbell is broken, and you're waiting for a UPS delivery while you have work to do in your home office.
- Problem: If the UPS arrives in between two polls, you may miss the delivery. If you shorten your polling intervals your desk work will become inefficient, since you spend a lot of time checking for the parcel that hasn't arrived yet.
- Polling entails high latency and is an inefficient way to handle events.

Interrupts

- Notify the system when events occur.
- Analogy: Doorbell (if it is working)
- When an interrupt occurs, the CPU alters its flow of instruction execution to handle the interrupt.
- Perfect for events that are time critical and/or infrequent.

Andre M. Maier
Dept. of Computer Science

Principle



Andre M. Maier
Dept. of Computer Science

Types of Interrupts

- Hardware Interrupts
  - Hardware signal from an external device that needs attention, e.g. key pressed on keyboard
- Software Interrupts
  - Exceptions in the processor, e.g. divide by zero exception
  - Software interrupt instructions (e.g. INT in x86 assembly)

Andre M. Maier
Dept. of Computer Science

Interrupt Vector Table (IVT)

- When an interrupt occurs, the CPU needs to “jump” to a certain address where the code to handle the interrupt, a.k.a. Interrupt Service Routine (ISR) resides.
- Interrupt Vector = Number that corresponds to the address of the interrupt handler
- IVT associates interrupt requests (sources) with interrupt vectors
- The IVT is called Interrupt Descriptor Table in x86 terminology
  - First 32 entries: Handlers for CPU-specific exceptions

Andre M. Maier
Dept. of Computer Science

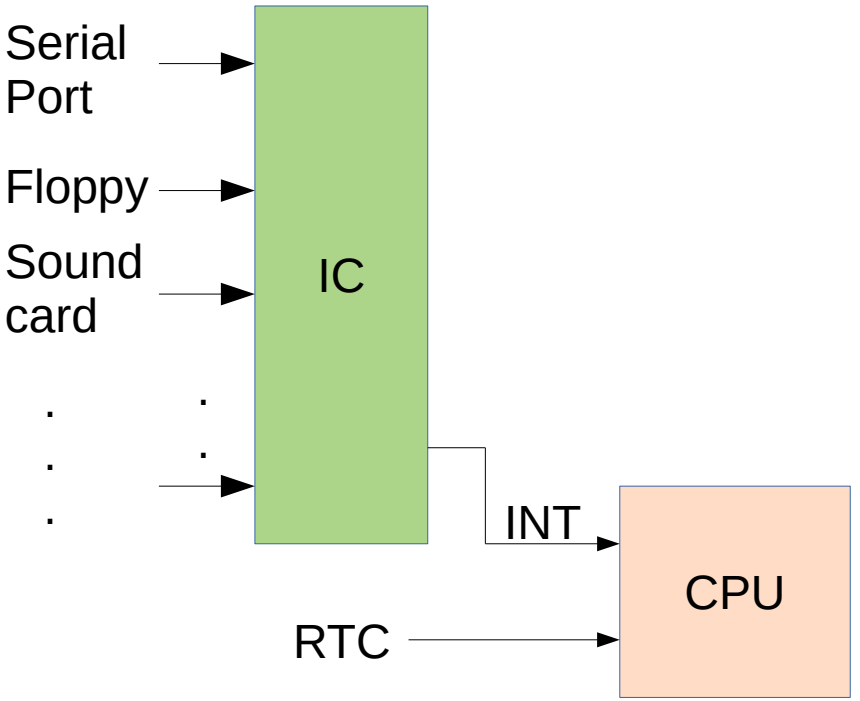
x86 Interrupt Vectors (Example)

Vector	Description
0	Divide Error
2	Non-Maskable Interrupt
3	Breakpoint Exception
6	Invalid Opcode
11	Segment Not Present
12	Stack-Segment Fault
13	General Protection Fault
14	Page Fault
18	Machine Check
32 … 255	User-defined Interrupts

Andre M. Maier
Dept. of Computer Science

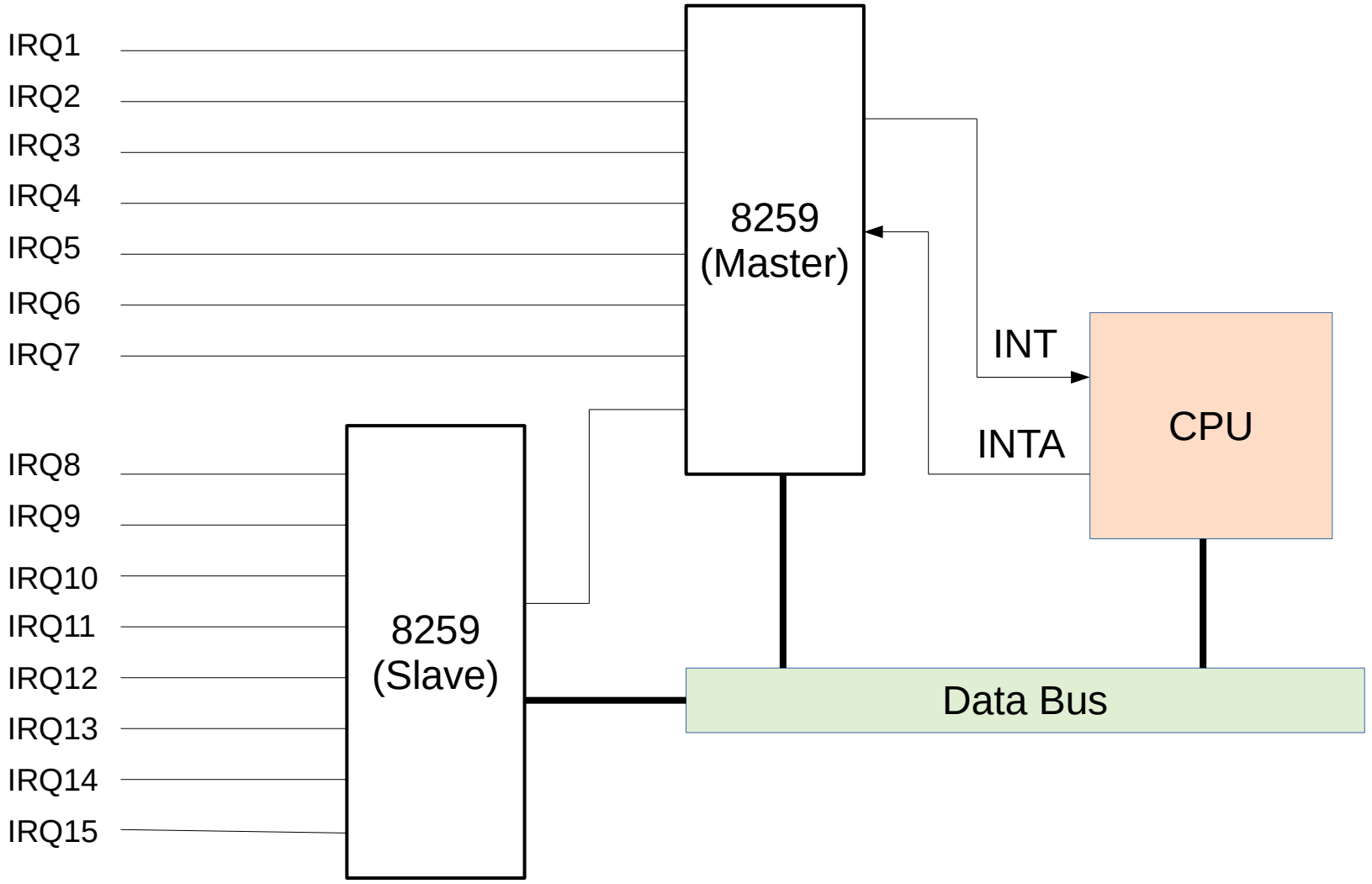
Hardware Interrupts

- CPUs have a limited number of interrupt pins (often no more than one).
- Devices are connected to Interrupt Controller
- The interrupt controller allows to combine multiple interrupt sources onto a limited number of CPU interrupt lines.



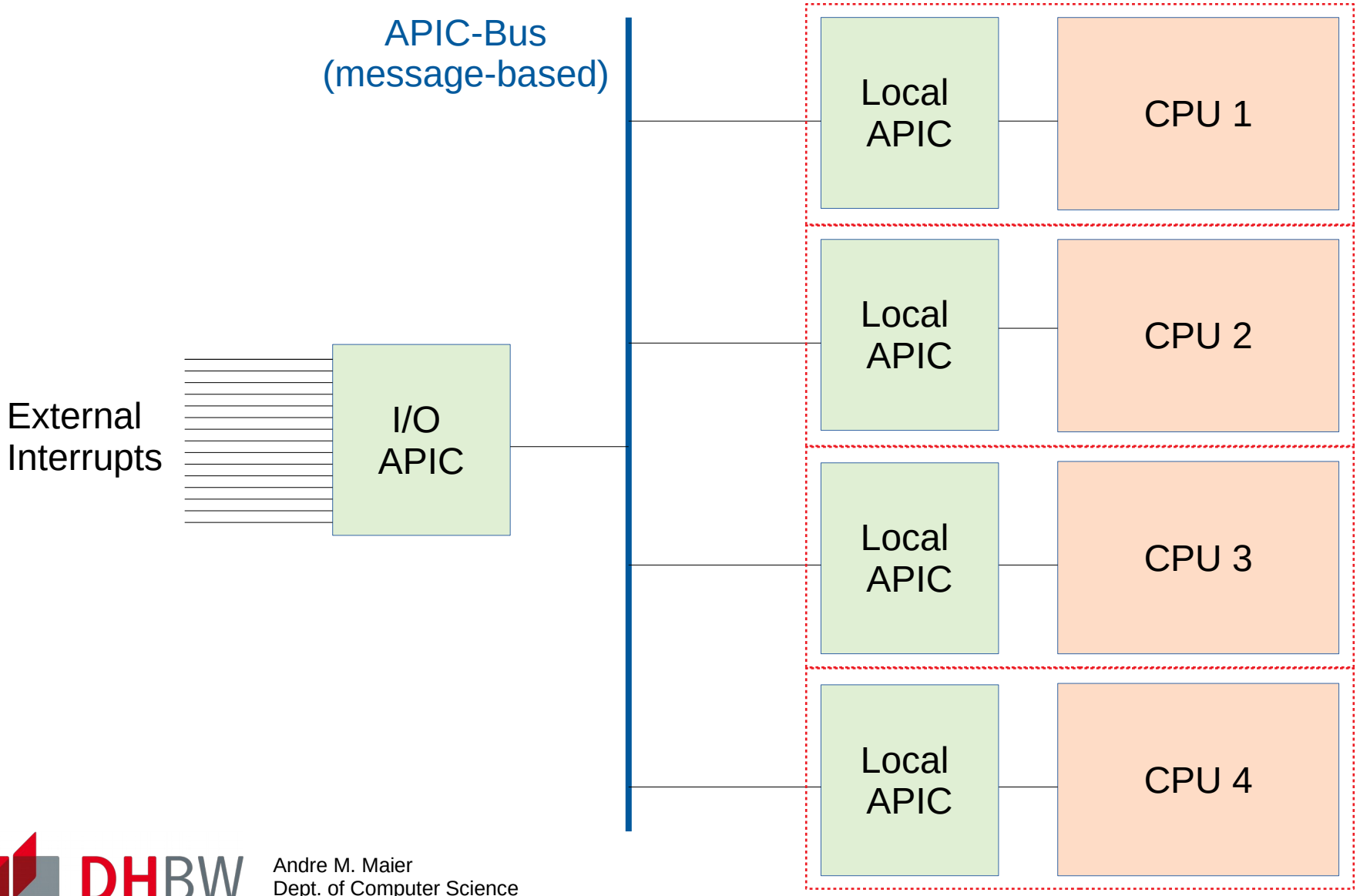
Andre M. Maier
Dept. of Computer Science

Interrupt Controller 8259 (Legacy)



Andre M. Maier
Dept. of Computer Science

Advanced Programmable IC (APIC)



Andre M. Maier
Dept. of Computer Science

Interrupt Triggers

- Level-triggered interrupts
  - CPU gets an interrupt as long as the IRQ line is active
  - Interrupt may persist even after ISR is completed
  - Active-high vs. active-low
- Edge-triggered interrupts
  - CPU gets only one interrupt when the IRQ line becomes active
  - Next interrupt will be triggered when the next edge occurs
  - Falling edge vs. rising edge

Andre M. Maier
Dept. of Computer Science

Spurious Interrupts

- Consider the following scenario:

  Some device activates a level-triggered IRQ line upon which the CPU gets notified of the fact that an interrupt is requested. The CPU acknowledges and waits for Interrupt Controller to send information on the source of the interrupt request. Just in this moment, the device deactivates the IRQ line.
- Solution: The interrupt controller tells the CPU that the interrupt source is **spurious**.
- Spurious interrupts should be handled with the lowest priority.

Interrupt Latency

- Latency is caused by
  - Hardware (APICs, CPUs)
  - Software (Operating System)
- In order to handle the interrupt, program state of the currently running program needs to be stored.
- Operating System will complete atomic operations before the interrupt can be handled.
- Nested interrupts improve responsiveness to IRQs.
- Real-time Operating Systems (RTOS) must guarantee a interrupt latency of less than a specified time.

Andre M. Maier
Dept. of Computer Science

Lab Exercises

- Examine the following directory and files
  - */proc/interrupts*
  - */proc/irq*
  - */sys/kernel/irq*
- Locate and monitor the hardware interrupt that occurs when a key is pressed on the keyboard.

Andre M. Maier
Dept. of Computer Science

Questions for Review

- What is the difference between polling and interrupt?
- Explain the two types of software interrupt.
- What is the purpose of an APIC?
- What is an Interrupt Vector?
- How can spurious interrupts occur?
- Why should interrupt handlers contain as little code as possible, especially in systems where nested interrupts are possible?

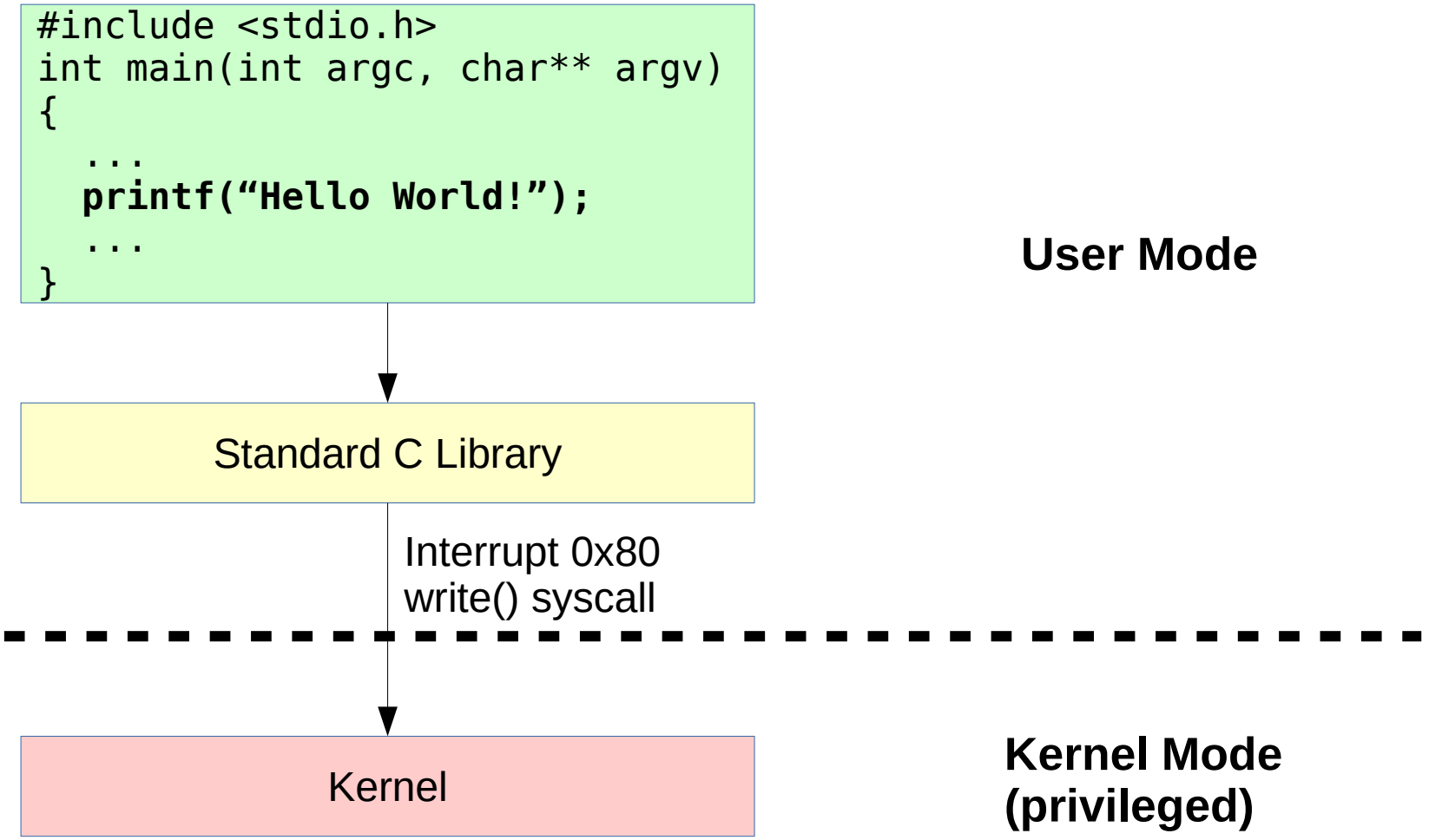
Andre M. Maier
Dept. of Computer Science

System Calls (1)

- System calls allow programs running in user space to request services from the kernel.
- Examples: fork(), write()
- System calls require the processor to switch from **user mode** into **kernel mode**.
- System calls are typically implemented by using a software interrupt (trap).
- In Linux, INT 128 (0x80) is used for system calls.

Andre M. Maier
Dept. of Computer Science

System Calls (2)



Andre M. Maier
Dept. of Computer Science

Lab Exercises

- Read the *syscalls* man page in Linux to find out what system calls are supported.
- Install and try the following programs
  - *strace*
  - *sysdig*
- Write a C program that invokes one or more system calls. Install and configure *auditd* to monitor system calls from your program.

Andre M. Maier
Dept. of Computer Science